

Simulation of Quantum Algorithms for Solving Machine Learning and Chemistry Problems

Steven Oud
SURFsara

September 20, 2019

Summary

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Fusce lobortis erat eget erat euismod dapibus. Interdum et malesuada fames ac ante ipsum primis in faucibus. Donec pharetra, magna ac tincidunt varius, neque odio lobortis velit, eget tincidunt lectus sapien ac velit. Donec maximus euismod arcu, at efficitur urna. Praesent viverra elementum elementum. Cras lacinia nisi eleifend sodales posuere. Fusce ut blandit purus, quis ornare est. Aenean varius purus lorem, a lobortis orci mollis pretium. Nulla facilisi. Nulla in neque orci. Integer dolor massa, ullamcorper nec sagittis id, mattis a lectus. Vivamus efficitur mi a elit feugiat facilisis. Ut lorem tortor, dignissim nec quam et, dapibus vehicula nibh. Quisque mollis enim quis odio interdum, ac blandit mauris maximus. Aenean ac dolor in augue accumsan porta.

Fusce at sodales turpis. Nullam pulvinar rhoncus diam eu consequat. Sed et ante ac augue rutrum cursus id id mi. Quisque aliquet at lectus eget condimentum. Quisque pharetra nulla vel vestibulum vulputate. Donec blandit lacus est, sed tempor nisi laoreet eget. Mauris maximus ultricies varius. Nullam id hendrerit ante, quis efficitur nibh. Pellentesque convallis dignissim dapibus. Nunc vehicula scelerisque posuere.

Contents

1	Introduction	3
2	Methods	3
3	Quantum Computation and Information	4
3.1	Quantum States	4
3.2	Quantum State Evolution	5
3.3	Entanglement	6
3.4	Quantum Speedup	6
3.5	Current State of Quantum Computing	6
4	Quantum Machine Learning	7
5	Quantum Chemistry	7
6	Implementations and Results	7
6.1	Quantum Neural Network	7
6.1.1	Binary Downsampled Classifier	7
6.1.2	One-vs-Rest Classifier	9
7	Conclusion	10
	References	10

1 Introduction

Quantum computing is one of the most promising technology developments of the coming years. Big companies like IBM (IBM, 2019), Google (Google AI, 2019), Intel (Intel, 2019), Microsoft (Microsoft, 2019), and countries like the USA and China are investing huge amounts of money into the development and building of meaningful working quantum computers (Raymer and Monroes, 2019; Kania and Costello, 2018). The development of practical quantum computers that can be used to solve real-world problems is driven by the expectation that for certain tasks, quantum computers can outperform classical computers (Preskill, 2012).

Already last year, SURFsara started a number of activities and collaborations in the field of quantum computing. The overall objective of SURFsara is to support Dutch researchers in taking an early and competitive advantage in quantum computing technologies and facilities while these become available. Like with any emerging compute technology, it needs early adopters in the scientific computing community to identify problems of practical interest that are suitable as proof-of-concept applications. In the context of the SURF Open Innovation Lab, SURFsara seeks to stimulate and support these advances in scientific research in close collaboration with research groups. Within this context SURFsara is interested in testing, benchmarking and creating good examples of quantum computing applications that can be used by the scientific community.

SURFsara HPC center supports various research institutes and universities of the Netherlands. The chemistry community is currently one of the largest; the machine learning community is probably the fastest growing one. Both of these fields are large areas of research within the quantum computing field, expecting improvements to be gained from them. As the main interest of SURFsara is to support the potential main use cases of quantum computers in the future, the tasks of this internship will be focused on quantum machine learning (QML) and quantum chemistry (QC).

This report will try to give an answer to the main question “*How can quantum algorithms be implemented using simulated quantum circuits to solve machine learning and chemistry problems?*” This is further expanded into the following sub questions:

1. What are current promising QML and QC algorithms?
2. What simulators are best suited for simulating QML and QC quantum circuits?
3. How can these quantum algorithms be integrated in current classical workflow of chemistry and machine learning applications?
4. How do these quantum algorithms perform (in regard to classical algorithms)?

This report is structured as follows. First, in Section 2, I describe how the research was conducted. Second, in Section 3, a short introduction to quantum computation and information is given. Third, in Section 4 and Section 5, research towards quantum machine learning and quantum chemistry is described. Finally, in Section 6, quantum algorithms are implemented using simulation and the results are discussed. The conclusion of the research is described in Section 7.

2 Methods

The research in this report was done mostly through desk research, with the help and advice of my colleagues. The first step was to get familiar with the QML and QC fields; what background knowledge is required, what is the current state of research and what are the next steps? To get started, several papers in promising areas of research were provided by my colleagues. From there, databases such as Google Scholar and arXiv were used to find further information about these areas. Search terms used include *quantum machine learning*, *quantum neural network*, *quantum support vector machine*, *variational quantum eigensolver*, *quantum phase estimation*, *quantum perceptron*, *quantum classifier*, *quantum machine learning library*, *quantum chemistry simulation* and *hybrid quantum optimization*.

3 Quantum Computation and Information

Quantum computers take advantage of quantum mechanical effects such as superposition and entanglement to solve certain problems faster than classical computers. The idea of a quantum computer was first proposed by Richard Feynman for simulating physics (Feynman, 1982). Since then the field has advanced a lot with algorithms such as Shor’s algorithm for factoring integers (Shor, 1999) and Grover’s search algorithm (Grover, 1996). These quantum algorithms show efficient solutions for problems that are considered hard for classical computers.

This section summarizes basic concepts of quantum computation and information theory required to understand how quantum computers can be used to speedup certain computational problems. This is a very high level overview of quantum computation and is far from a complete introduction to the field. For a more complete introduction to quantum computation and information, we refer to the book by Nielsen and Chuang, 2011.

3.1 Quantum States

The basic and smallest unit of quantum information is the quantum bit, or *qubit*. A qubit is a two-state quantum-mechanical system, which can be represented for example in the spin of an electron. A qubit can be in a state of an orthonormal basis $\{|0\rangle, |1\rangle\}$ (corresponding to spin up and spin down for the electron example), but it can also be in a linear combination, or *superposition*, of states. The state of a qubit (the so-called *wave function*) can be described as following:

$$|\psi\rangle = c_0 |0\rangle + c_1 |1\rangle. \quad (1)$$

Quantum states are denoted using Dirac notation $|\cdot\rangle$, which is a column vector in \mathbb{C}^n . Here the coefficients $\{c_0, c_1\}$ are complex valued numbers called *probability amplitudes*. When we measure a quantum state, it collapses probabilistically to one of the basis states. The norm square $|c_0|^2$ gives us the probability of finding the particle in state $|0\rangle$, and $|c_1|^2$ gives us the probability of finding the particle in state $|1\rangle$. As we are dealing with probabilities, the quantum state should be normalized: $\sum_{i=0} |c_i|^2 = 1$. As an example we have the arbitrary state:

$$|\psi\rangle = \frac{i}{2} |0\rangle - \frac{\sqrt{3}}{2} |1\rangle. \quad (2)$$

After measuring this state, we have a $|i/2|^2 = 1/4$ chance of the system being in the state $|0\rangle$, and a $|\sqrt{3}/2|^2 = 3/4$ chance of being in the state $|1\rangle$. The laws of quantum mechanics greatly restrict our access to information stored in quantum states. We cannot access the amplitudes of a state, other than by preparing and sampling a state many times to get an approximation. When we measure a state, it collapses to a basis state $|j\rangle$ with probability $|c_j|^2$. If we measure again immediately after the first measure, we get the same result 100% of the time. So if we measure $|1\rangle$ and measure again immediately after, we will see $|1\rangle$ again. Measuring a quantum state collapses the wave function and destroys state information.

We define the computational basis states $\{|0\rangle, |1\rangle\}$ as following:

$$|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \quad |1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}. \quad (3)$$

This idea generalizes to multi-qubit systems. A multi-qubit system with n states $\{|\psi_1\rangle, |\psi_2\rangle, \dots, |\psi_n\rangle\}$ can be represented using the *Kronecker product*¹:

$$|\psi\rangle = |\psi_1\rangle \otimes |\psi_2\rangle \otimes \dots \otimes |\psi_n\rangle, \quad (4)$$

resulting in a $N = 2^n$ dimensional state $|\psi\rangle$. For example, a three qubit state lives in a 2^3 -dimensional Hilbert space spanned by computational basis states $\{|000\rangle, |001\rangle, |010\rangle, \dots, |111\rangle\}$:

¹Entangled states are an exception to this, which will be further discussed in Section 3.3

$$\begin{aligned}
|\psi\rangle &= c_0 |000\rangle + c_1 |001\rangle + c_2 |010\rangle + \cdots + c_{N-1} |111\rangle \\
&= c_0 \begin{pmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix} + c_1 \begin{pmatrix} 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix} + c_2 \begin{pmatrix} 0 \\ 0 \\ 1 \\ \vdots \\ 0 \end{pmatrix} + \cdots + c_{N-1} \begin{pmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ 1 \end{pmatrix} = \begin{pmatrix} c_0 \\ c_1 \\ c_2 \\ \vdots \\ c_{N-1} \end{pmatrix}.
\end{aligned} \tag{5}$$

3.2 Quantum State Evolution

In classical computers, we manipulate bits through logical gates. Quantum computers use quantum gates, which transform one quantum state to another through an operator U . These operators must be unitary. That is, they must preserve the norm of the vector and be reversible: $U^\dagger U = U U^\dagger = I$ (where † is the complex conjugate and I the identity matrix). Single qubit states can be thought of as a vector on the surface of a sphere (the Bloch sphere). A unitary operation can then be thought of as rotations around the x , y and z axes of the sphere (Figure 1).

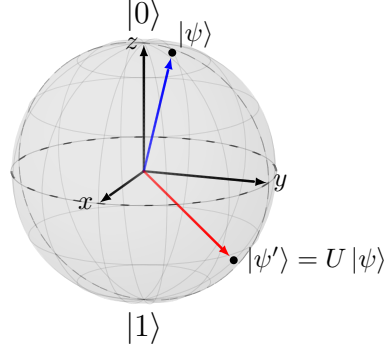


Figure 1: Arbitrary transformation of state $|\psi\rangle$ by operator U visualized on the Bloch sphere.

As an example, the X gate is equivalent to the classical NOT: $X|0\rangle = |1\rangle$ and $X|1\rangle = |0\rangle$. It is known as one of the three Pauli matrices used in quantum mechanics:

$$\sigma_1 = \sigma_x = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad \sigma_2 = \sigma_y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}, \quad \sigma_3 = \sigma_z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}. \tag{6}$$

Another frequently used gate is the Hadamard gate, which produces an equal superposition:

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}. \tag{7}$$

Which acts on the computational basis states as following:

$$\begin{aligned}
H|0\rangle &= \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \end{pmatrix} \\
&= \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle), \\
H|1\rangle &= \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ -1 \end{pmatrix} \\
&= \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle).
\end{aligned} \tag{8}$$

Applying the Hadamard gate again on the equal superposition created above is an example of destructive interference:

$$\begin{aligned} H \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) &= |0\rangle, \\ H \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) &= |1\rangle. \end{aligned} \tag{9}$$

We find that applying the Hadamard twice is the same as doing nothing, or more formally, we say the Hadamard gate is *Hermitian*: $H = H^\dagger$.

3.3 Entanglement

There also exist multi-qubit gates, which are required for universal quantum computing. These gates introduce the phenomena entanglement. Consider the following state:

$$|\Phi^+\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle). \tag{10}$$

Measuring the first qubit gives state $|0\rangle$ with probability $1/2$, and state $|1\rangle$ with probability $1/2$. However, the measurement immediately collapsed the whole state to either $|00\rangle$ or $|11\rangle$. So by measuring the first qubit, we also know the state of the second qubit. The individual states are related to each other, and this relation is called entanglement. More formally, two qubits are entangled if and only if the state of those two qubits cannot be expressed as two individual states. For the entangled state $|\Phi^+\rangle$, let's assume there exist amplitudes $\{c_0, c_1, c'_0, c'_1\}$ such that:

$$|\Phi^+\rangle = (c_0|0\rangle + c_1|1\rangle) \otimes (c'_0|0\rangle + c'_1|1\rangle) = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle). \tag{11}$$

However, this would imply that $c_0c'_0 = c_1c'_1 = 1/\sqrt{2}$ and $c_0c'_1 = c_1c'_0 = 0$, which does not have a solution.

The remarkable thing about entanglement is that it works over any distance. That is, given the state $|\Phi^+\rangle$, one qubit could be located in another galaxy while the other qubit is located on earth. When we measure the qubit on earth, we know that the qubit in the other galaxy is in the same state as we measured. This does not allow for faster than light communication, as a classical channel is still required to communicate results between the observers.

3.4 Quantum Speedup

Quantum algorithms promise to provide a speedup over classical algorithms by “abusing” quantum phenomena such as superposition, interference, exponential state space and entanglement. They usually offer a quadratic or exponential speedup over their classical counterpart. Aaronson, 2011 probably described it best: “The goal in quantum computing is to choreograph a computation so that the amplitudes leading to wrong answers cancel each other out, while the amplitudes leading to right answers reinforce.”

The most famous example of a quantum speedup is probably Shor's algorithm for factoring integers and computing discrete logarithms, which was introduced in 1994. Its implications are huge for modern cryptography algorithms such as RSA, which depend on the fact that factoring integers is a hard problem on classical computers. Shor showed that this can be done in polynomial time on a quantum computer (Shor, 1999), providing an exponential speedup over classical algorithms. It has been estimated that a 2048-bit RSA integer could be factored in eight hours using 20 million noisy qubits (Gidney and Ekerå, 2019).

3.5 Current State of Quantum Computing

It is important to note that quantum computers are still at the experimental stage, and a lot of results are theoretical. For example, the highest number factored by using Shor's algorithm

on an actual quantum computer thus far is 21 (Martín-López et al., 2012). Furthermore, quantum supremacy, which is the potential ability of a quantum computer to solve problems that classical computers practically cannot (Preskill, 2012), has not been achieved yet as of writing this. Current and near future quantum computers, often referred to as Noisy Intermediate-Scale Quantum (NISQ) devices, will have about 50-100 qubits and may be able to achieve quantum supremacy (Preskill, 2018). However, they suffer from noise, which greatly limits their coherence time and thus usefulness. Quantum error correcting codes exist which can be used to protect from noise, but these require a high amount of qubits per logical qubit.

Research has adapted to the limitations of NISQ devices, and hybrid quantum/classical algorithms have become an important area of research. Examples of such algorithms are the quantum approximate optimization algorithm (QAOA) for optimization problems (Farhi, Goldstone, and Gutmann, 2014), and the variational quantum eigensolver (VQE) for finding eigenvalues of a Hamiltonian (Peruzzo et al., 2014). These hybrid algorithms typically involve a small quantum subroutine run inside of a classical optimization loop, removing the need for a large-scale, coherent quantum computer.

This research is focused on universal gate model quantum computers. There exist special purpose quantum computers called quantum annealers, which have achieved qubit counts up to 2000 (D-Wave, 2017). These quantum annealers are restricted to solving optimization problems, of which its usefulness has been doubted (Shin, Smith, Smolin, and Vazirani, 2014; Aaronson, 2013; Rønnow et al., 2014).

4 Quantum Machine Learning

Quantum machine learning is still in its very first stage of development, and lacks a widely recognized definition. However, there have been promising developments in the recent years.

5 Quantum Chemistry

6 Implementations and Results

This section describes how QML and QC algorithms were implemented, tested and benchmarked. The experimental results described were achieved by simulating quantum circuits using SURFsara’s Lisa computing system (SURFsara, 2019).

6.1 Quantum Neural Network

A quantum neural network inspired by the work of Farhi and Neven, 2018 was implemented for classifying handwritten images. The famous MNIST database of handwritten digits (LeCun and Cortes, 2010) was used for this (Figure 2). For initial experiment, the images of the data set were downsampled to a lower dimension and limited to binary classification of digits 1 and 8 to simplify the problem and reduce resource requirements. After a successful initial experiment, the experiment is extended to multi-class classification of the original 28×28 data set.

6.1.1 Binary Downsampled Classifier

The problem of supervised classification in machine learning is defined as following: given a training set of data whose category is known, train a model which is able to identify to which category new data belongs. A simple example that we will tackle in this section is recognizing if a handwritten digit is a 1 or an 8.

We implement a quantum neural network to classify 8×8 images of handwritten digits (Figure 2b). It uses a quantum circuit together with a classical optimization algorithm to find the optimal parameters for the quantum circuit, often known as a variational algorithm. An overview

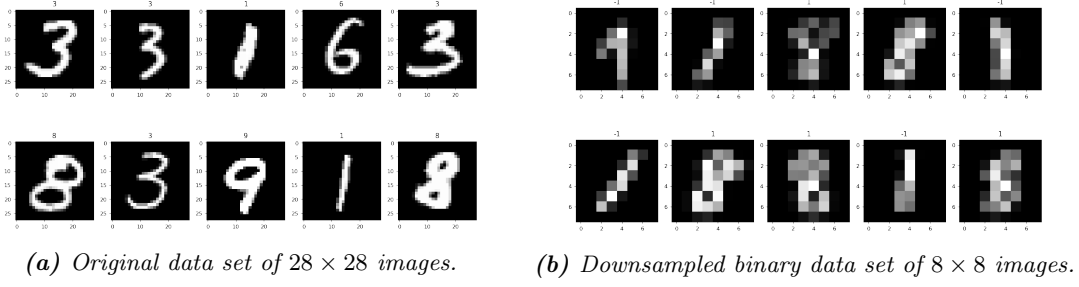


Figure 2: Samples of MNIST data set used for quantum neural network binary classification.

of the quantum circuit used in the experiment can be found in Figure 3. The variational quantum neural network works as following:

1. Prepare data sample \vec{x} in the amplitudes of a quantum state $|x\rangle$ using for example qRAM (Giovannetti, Lloyd, and Maccone, 2008).
2. Prepare a readout qubit in the state $|1\rangle$, giving $|x, 1\rangle$.
3. Apply L layers of parameterized unitaries: $U(\vec{\theta}) |x, 1\rangle$, where $U(\vec{\theta}) = U(\vec{\theta}_L)U(\vec{\theta}_{L-1}) \dots U(\vec{\theta}_1)$.
4. Measure expectation value $\langle Z \rangle = \langle x, 1 | U(\vec{\theta})^\dagger Z_n U(\vec{\theta}) |x, 1\rangle$ of readout qubit.
5. Calculate and minimize the loss $L(\vec{\theta}, \langle Z \rangle)$ through a classical optimization algorithm.
6. Repeat until convergence.

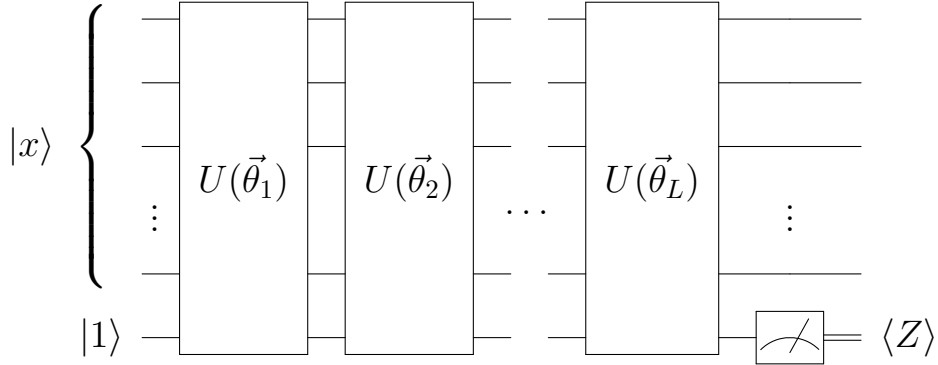


Figure 3: General circuit of the quantum neural network classifier. The network consist of L layers with each layer implementing a parameterized unitary $U(\vec{\theta}_i)$ whose parameters are optimized using a classical optimization algorithm. At the end of the circuit, the expectation value $\langle Z \rangle$ of the readout qubit is measured and the predicted class is decided by $\text{sgn}(\langle Z \rangle)$. The decomposition of a layer is shown in Figure 4.

We trained a quantum neural network using a training set of 2014 images and a test set of 843 images. As optimizer, we used the stochastic mini-batch gradient descent with a learning rate of 0.4 and a batch size of 32. After training for 20 epochs with 2 layers, it managed to reach 91.1% accuracy (Figure 5) without any hyperparameter tuning or circuit optimization. The goal of this experiment is not to compete with classical state-of-the-art neural networks, which are

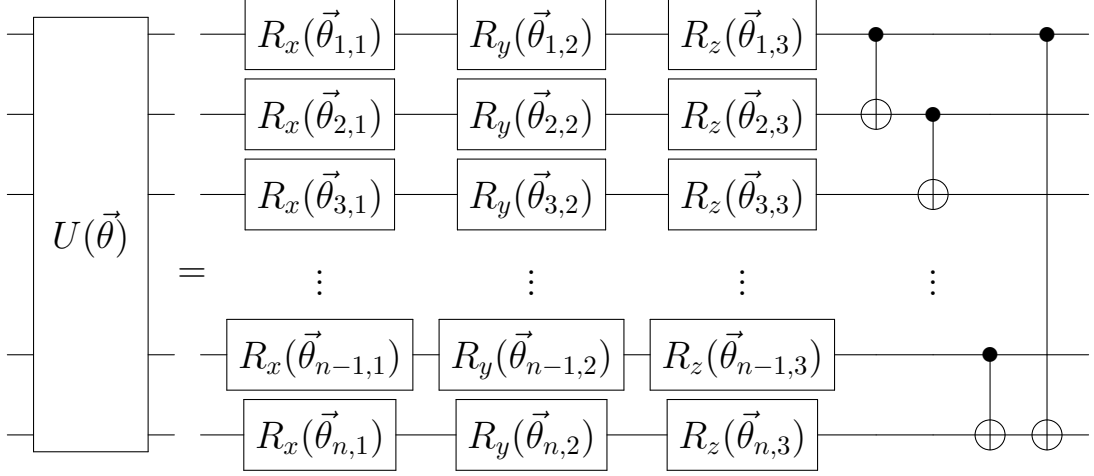


Figure 4: Decomposition of a layer $U(\vec{\theta})$. The amount of trainable parameters for a QNN with n qubits and L layers is $3n \cdot L$. The amount of qubits required is decided by the dimension of the data: $n = \lceil \log_2 N \rceil + 1$, where N is the input dimension. The downsampled MNIST classifier requires $\lceil \log_2(8 \cdot 8) \rceil + 1 = 7$ qubits.

capable of achieving an error percentage of only 0.23% (Cireřan, Meier, and Schmidhuber, 2012)². Rather, it is meant to demonstrate the possibility of using quantum computers for machine learning problems and hint that the universal approximation theorem (Csaji, 2001) applies to quantum neural networks. It shows the potential, given we solve the issues surrounding qRAM (Aaronson, 2015), to use quantum computers to process (quantum) data which classical computers cannot process efficiently.

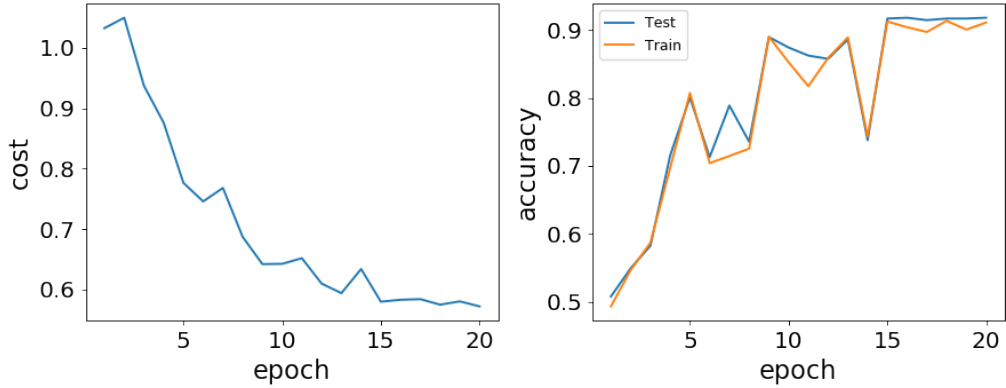


Figure 5: Quantum neural network performance on binary classification.

6.1.2 One-vs-Rest Classifier

To make a step into the direction of multi-class classification using a QNN, we first consider the problem of one-vs-rest classification. We use the original 28×28 MNIST data set to train a QNN

²Which is not to say quantum neural networks can't compete with classical neural networks. In fact, Perez-Salinas, Cervera-Lierta, Gil-Fuster, and Latorre, 2019 show a single-qubit classifier to be at least comparable with classical methods.

to detect if a handwritten digit is 0 or not 0 (Figure 6). To encode a 28×28 -dimensional image we need $\lceil \log_2(28 \cdot 28) \rceil = 10$ qubits, plus a readout qubit, for 11 qubits total.

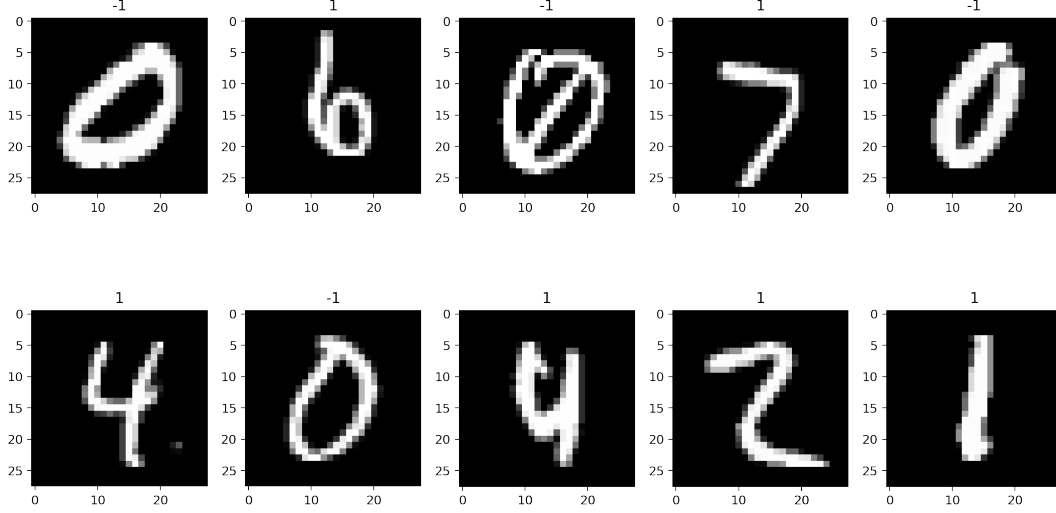


Figure 6: Sample of data set used for one-vs-rest classification. The classifier has to decide if a handwritten digit is 0 or not 0.

7 Conclusion

References

- Aaronson, S. (2011). *Quantum computing promises new insights, not just supermachines*. Retrieved 2019-09-18, from <https://www.nytimes.com/2011/12/06/science/scott-aaronson-quantum-computing-promises-new-insights.html>
- Aaronson, S. (2013). *D-Wave: truth finally starts to emerge*. Retrieved 2019-09-19, from <https://www.scottaaronson.com/blog/?p=1400>
- Aaronson, S. (2015). *Read the fine print*. *Nature Physics*, 11(4), 291.
- Cireřan, D., Meier, U., & Schmidhuber, J. (2012). *Multi-column deep neural networks for image classification*. eprint: arXiv:1202.2745
- Csaji, B. C. (2001). *Approximation with artificial neural networks*. *Faculty of Sciences, Etvs Lornd University, Hungary*, 24, 48.
- D-Wave. (2017). *D-Wave announces D-Wave 2000Q quantum computer and first system order*. Retrieved 2019-09-10, from <https://www.dwavesys.com/press-releases/d-wave%2%a0announces%2%a0d-wave-2000q-quantum-computer-and-first-system-order>
- Farhi, E., Goldstone, J., & Gutmann, S. (2014). *A quantum approximate optimization algorithm*. eprint: arXiv:1411.4028
- Farhi, E. & Neven, H. (2018). *Classification with quantum neural networks on near term processors*.
- Feynman, R. P. (1982). *Simulating physics with computers*. *International journal of theoretical physics*, 21(6), 467–488.
- Gidney, C. & Eker, M. (2019). *How to factor 2048 bit RSA integers in 8 hours using 20 million noisy qubits*. eprint: arXiv:1905.09749
- Giovannetti, V., Lloyd, S., & Maccone, L. (2008). *Quantum random access memory*. *Physical review letters*, 100(16), 160501.
- Google AI. (2019). *Quantum*. Retrieved 2019-09-11, from <https://ai.google/research/teams/applied-science/quantum/>
- Grover, L. K. (1996). *A fast quantum mechanical algorithm for database search*. *arXiv preprint quant-ph/9605043*.
- IBM. (2019). *Quantum computing*. Retrieved 2019-09-11, from <https://www.ibm.com/quantum-computing/>

- Intel. (2019). *Quantum computing*. Retrieved 2019-09-11, from <https://www.intel.com/content/www/us/en/research/quantum-computing.html>
- Kania, E. B. & Costello, J. (2018). *Quantum hegemony? China's ambitions and the challenge to U.S. innovation leadership frontier*. Retrieved 2019-09-16, from <https://www.cnas.org/publications/reports/quantum-hegemony>
- LeCun, Y. & Cortes, C. (2010). *MNIST handwritten digit database*. Retrieved from <http://yann.lecun.com/exdb/mnist/>
- Martín-López, E., Laing, A., Lawson, T., Alvarez, R., Zhou, X.-Q., & O'brien, J. L. (2012). *Experimental realization of Shor's quantum factoring algorithm using qubit recycling*. *Nature Photonics*, 6(11), 773.
- Microsoft. (2019). *Quantum computing*. Retrieved 2019-09-11, from <https://www.microsoft.com/en-us/quantum/>
- Nielsen, M. A. & Chuang, I. L. (2011). *Quantum computation and quantum information: 10th anniversary edition* (10th). New York, NY, USA: Cambridge University Press.
- Pérez-Salinas, A., Cervera-Lierta, A., Gil-Fuster, E., & Latorre, J. I. (2019). *Data re-uploading for a universal quantum classifier*. *arXiv preprint arXiv:1907.02085*.
- Peruzzo, A., McClean, J., Shadbolt, P., Yung, M.-H., Zhou, X.-Q., Love, P. J., . . . O'brien, J. L. (2014). *A variational eigenvalue solver on a photonic quantum processor*. *Nature communications*, 5, 4213.
- Preskill, J. (2012). *Quantum computing and the entanglement frontier*. eprint: arXiv:1203.5813
- Preskill, J. (2018). *Quantum computing in the NISQ era and beyond*. *Quantum*, 2, 79.
- Raymer, M. G. & Monroes, C. (2019). *The US national quantum initiative*. *Quantum Science and Technology*, 4(2), 020504. doi:10.1088/2058-9565/ab0441
- Rønnow, T. F., Wang, Z., Job, J., Boixo, S., Isakov, S. V., Wecker, D., . . . Troyer, M. (2014). *Defining and detecting quantum speedup*. *Science*, 345(6195), 420–424.
- Shin, S. W., Smith, G., Smolin, J. A., & Vazirani, U. (2014). *How "quantum" is the d-wave machine?*
- Shor, P. W. (1999). *Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer*. *SIAM review*, 41(2), 303–332.
- SURFsara. (2019). *The Lisa system*. Retrieved 2019-09-20, from <https://userinfo.surfsara.nl/systems/lisa>